

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-301856

(43)Date of publication of application : 13.11.1998

(51)Int.Cl.

G06F 12/14  
G06F 12/00

(21)Application number : 10-045268

(71)Applicant : FUJITSU LTD

(22)Date of filing : 26.02.1998

(72)Inventor : SASAKI TAKAOKI  
YAMANAKA YUSUKE

(30)Priority

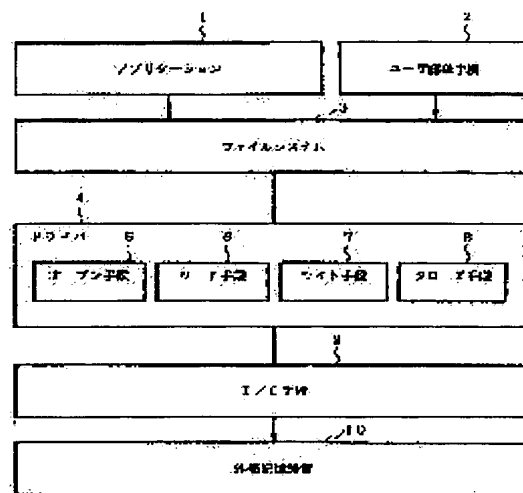
Priority number : 09 46724 Priority date : 28.02.1997 Priority country : JP

## (54) FILE ACCESS SYSTEM AND RECORDING MEDIUM

(57)Abstract:

PROBLEM TO BE SOLVED: To easily and partially access a deciphered file at a high speed and to raise a security level by taking out a part with an access request from a ciphered file by a read means or a write means, deciphering it or deciphering and writing it, then ciphering it and writing it back.

SOLUTION: When data read from a pertinent file in an external storage device 10 by an I/O means 9 corresponding to a read request from an application 1 are ciphered, the read means 6 returns the deciphered data of the same size to the application 1 of a read request origin or the like. Also, when the data read from the pertinent file in the external storage device 10 by the I/O means 9 corresponding to a write request from the application 1 are ciphered, the write means performs subscribing to the deciphered data of the same size and writes them back to the pertinent file in the external storage device 10.



## LEGAL STATUS

[Date of request for examination]

12.11.1999

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2000 Japanese Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-301856

(43) 公開日 平成10年(1998)11月13日

(51) Int.Cl.<sup>6</sup>

G 0 6 F 12/14  
12/00

識別記号

3 2 0  
5 3 7

F I

G 0 6 F 12/14  
12/00

3 2 0 B  
5 3 7 H

審査請求 未請求 請求項の数 6 O L (全 13 頁)

(21) 出願番号 特願平10-45268

(22) 出願日 平成10年(1998) 2月26日

(31) 優先権主張番号 特願平9-46724

(32) 優先日 平 9 (1997) 2月28日

(33) 優先権主張国 日本 (J P)

(71) 出願人 000005223

富士通株式会社

神奈川県川崎市中原区上小田中4丁目1番  
1号

(72) 発明者 佐々木 孝興

東京都港区芝浦四丁目15番33号 株式会社  
富士通ビー・エス・シー内

(72) 発明者 山中 祐介

東京都港区芝浦四丁目15番33号 株式会社  
富士通ビー・エス・シー内

(74) 代理人 弁理士 岡田 守弘

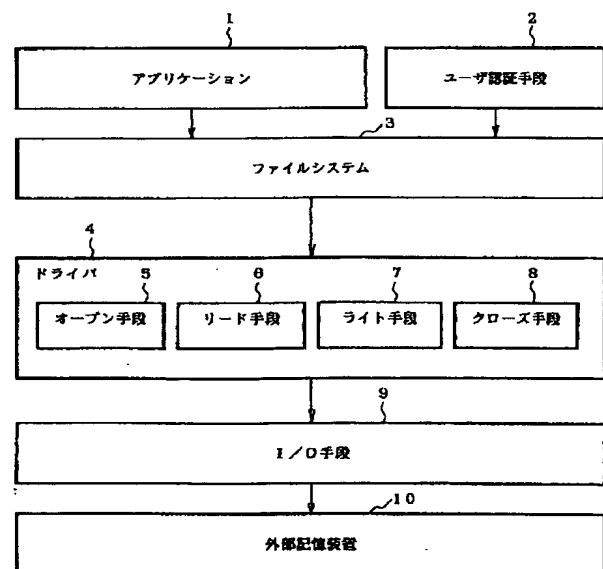
(54) 【発明の名称】 ファイルアクセスシステムおよび記録媒体

(57) 【要約】

【課題】 本発明は、オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されたリード/ライト命令に対してアクセスするファイルアクセスシステムおよび記録媒体に関し、アクセス依頼に対応してリード手段あるいはライト手段が暗号化されているファイルからアクセス依頼のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻し、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行い、セキュリティレベルを高めることを目的とする。

【解決手段】 アプリケーションから発行される前記リード命令に対して、リード処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、リード命令の処理対象のデータを復号化して、リード命令の発行元に返すリード手段を備えたるように構成する。

本発明のシステムブロック図



**【特許請求の範囲】**

【請求項1】オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されたリード命令に対して、実際のファイル内の指定されたデータをリード命令を発行したアプリケーションに返す、ファイルアクセスシステムにおいて、

アプリケーションから発行される前記リード命令に対して、リード処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、リード命令の処理対象のデータを復号化して、リード命令の発行元に返すリード手段を備えたことを特徴とするファイルアクセスシステム。

【請求項2】オープン命令が発行されたファイルの一部のデータ領域に対する、アプリケーションから発行されるデータのライト命令に対して、処理対象のファイルの一部のデータ領域に対して、指定されたデータを書き込むファイルアクセスシステムにおいて、

アプリケーションから発行される前記ライト命令に対して、ライト命令の処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、ライト命令の処理対象のファイルの一部のデータ領域を含み、かつ、暗号化の単位の長さのデータを読み出して復号化し、ライト命令で指定されたデータを該復号化したデータに上書きし、暗号化して、処理対象のファイルのデータ領域に格納するライト手段を備えたことを特徴とするファイルアクセスシステム。

【請求項3】前記ファイルから読み出した暗号化されていたデータを復号化して同一サイズのデータにし、あるいは暗号化されていないデータを暗号化して同一サイズのデータにしてファイルに書き込むことを特徴とする請求項1あるいは請求項2記載のファイルアクセスシステム。

【請求項4】ユーザ毎あるいは前記ファイル毎に前記リード手段による復号化あるいは前記ライト手段による復号化／暗号化の可あるいは不可を登録したことを特徴とする請求項1ないし請求項3記載のいずれかのファイルアクセスシステム。

【請求項5】コンピュータを動作させて、オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されるリード命令に対して、リード処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、リード命令の処理対象のデータを復号化して、リード命令の発行元に返すリード手段を機能させるプログラムを格納した記憶媒体。

【請求項6】コンピュータを動作させて、オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されるデータのライト命令に対して、ライト命令の処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化さ

れているファイルのうち、ライト命令の処理対象のファイルの一部のデータ領域を含み、かつ、暗号化の単位の長さのデータを読み出して復号化し、ライト命令で指定されたデータを該復号化したデータに上書きし、暗号化して、処理対象のファイルのデータ領域に格納するライト手段を機能させるプログラムを格納した記憶媒体。

**【発明の詳細な説明】****【0001】**

【発明の属する技術分野】本発明は、ファイルにアクセスするファイルアクセスシステムおよび記録媒体に関するものである。

**【0002】**

【従来の技術】従来、アプリケーションなどからアクセス依頼のあったデータについて、外部記憶装置に格納されているファイルをアクセスしている。この際、ファイルが暗号化されている場合には、ファイル内のデータを作業ファイルに一旦取り出して復号化した後、当該作業ファイルをアクセスしてアクセス処理終了後に暗号化して外部記憶装置のファイル内に書き戻すようにしている。

**【0003】**

【発明が解決しようとする課題】上述したように、従来は、外部記憶装置中のファイルが暗号化されていた場合には、当該ファイル内の暗号化されているデータを一旦作業ファイルに取り出して復号化した後、当該作業ファイルにアクセスしてアクセス処理を行い、一連の処理が終了した後、暗号化して元のファイルに書き戻すようにしていたため、アクセス依頼に対応してファイルが暗号化されていると作業ファイル作成や復号化して作業ファイルに格納などの処理が必要となり、迅速にファイルにアクセスできないという問題があった。また、復号化されたデータが作業ファイルに存在しセキュリティに欠けるという問題もあった。

【0004】本発明は、これらの問題を解決するため、アクセス依頼に対応してリード手段あるいはライト手段が暗号化されているファイルからアクセス依頼のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻し、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行い、セキュリティレベルを高めることを目的としている。

**【0005】**

【課題を解決するための手段】図1を参照して課題を解決するための手段を説明する。図1において、アプリケーション1は、アクセス要求を行うものである。このアプリケーションは既存のアプリケーションであり、このシステムのためにソースコードの変更を行う必要がない。

【0006】ファイルシステム3は、アプリケーション1からのアクセス要求に対応してドライバ4に指示してアクセス処理を行うものである。ドライバ4は、ファイ

ルの各種処理を行うものであって、ここでは、ファイルのリード処理を行うリード手段6、ファイルへのデータのライト処理を行うライト手段7などから構成されるものである。

【0007】I/O手段9は、外部記憶装置10のファイルからデータを読み出したり、書き戻したりなどするものである。次に、動作を説明する。

【0008】リード手段6がアプリケーション1からのリード要求に対応して、リード要求のあったデータについて、I/O手段9によって外部記憶装置10中の該当ファイルから読み出させたデータが暗号化されていたときは復号化したデータをリード要求元のアプリケーション1などに返し、一方、暗号化されていなかったときはそのままのデータをリード要求元のアプリケーション1などに返すようにしている。

【0009】また、ライト手段7がアプリケーション1からのライト要求に対応して、ライト要求のあったデータについて、I/O手段9によって外部記憶装置10中の該当ファイルから読み出させたデータが暗号化されていたときは復号化したデータに対して上書きした後、一方、暗号化されていなかったときはそのままのデータに対して上書きした後、I/O手段9によって外部記憶装置10中の該当ファイルに書き戻させるようにしている。

【0010】これらの際に、リード手段6がファイルから読み出した暗号化されていたデータを復号化して同一サイズのデータにし、あるいはライト手段7が暗号化されていないデータを暗号化して同一サイズのデータにしてファイルに書き込むようにしている。

【0011】また、ユーザ毎あるいはファイル毎にリード手段6による復号化あるいはライト手段7による復号化/暗号化の可あるいは不可を登録するようにしている。従って、アクセス依頼に対応してリード手段6あるいはライト手段7が暗号化されているファイルからアクセス要求のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻すことにより、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行い、セキュリティレベルを高めることが可能となる。

【0012】

【発明の実施の形態】次に、図1から図9を用いて本発明の実施の形態および動作を順次詳細に説明する。ここで、図示外の記録媒体あるいは外部記憶装置であるハードディスク装置などから読み出したプログラム、または回線を介してセンタから転送を受けたプログラムを主記憶にローディングして起動し、以下に説明する各種処理を行うようにしている。

【0013】図1は、本発明のシステムブロック図を示す。図1において、ユーザ認証手段2は、ユーザの認証（資格検査）を行うものである（図3を用いて後述す

る）。

【0014】ドライバ4は、ファイルのオープン処理、リード処理、ライト処理、クローズ処理を行うものであって、オープン手段5、リード手段6、ライト手段7、クローズ手段8などから構成されるものである。

【0015】オープン手段5は、外部記憶装置10中のファイルをオープンするものである（図4を用いて後述する）。リード手段6は、外部記憶装置10中のファイルのデータを読み出させて暗号化されているときは同一サイズのデータに復号化してリード要求元に返したり、暗号化されていなかったときはそのままデータをリード要求元に返したりなどするものである（図5を用いて後述する）。

【0016】ライト手段7は、外部記憶装置10中のファイルのデータを読みださせて暗号化されているときは同一サイズのデータに復号化してライト要求のあったデータを上書きし、一方、暗号化されていなかったときはデータにライト要求のあったデータを上書きし、元のファイルに書き戻させたりなどするものである（図6を用いて後述する）。

【0017】クローズ手段8は、外部記憶装置10中のファイルをクローズするものである（図7を用いて後述する）。次に、図2のフローチャートに示す順序に従い、図1の初期処理を詳細に説明する。

【0018】図2は、本発明の初期処理フローチャートを示す。図2において、S1は、OS（オペレーティングシステム）がドライバ確認コマンドを送信する。

【0019】S2は、S1で送信されたドライバ確認コマンドを受信したドライバ（#1）4が処理処理OKか判別する。OKのときはOK応答をOSに返してS4でOSにドライバ（#1）として登録する。一方、NGの場合には、NG応答をOSに返してS3でエラー処理（ドライバ（#1）がエラーとして登録などの処理）する。

【0020】S5は、S4でドライバ（#1）としてOSに登録されたので、OSが初期処理の指示をドライバ（#1）に通知する。S6は、S5の初期処理の通知に対応して、ドライバ（#1）がファイルシステムのイベントをフックする要求を発行する。具体的には、フック時にコールされるプログラムのアドレス（例えば“FS Hook（）；”など）をOSに指定する。

【0021】S7は、S6の要求に対応して、OSがフックアドレスなどを登録した後、その応答を返す。これは、例えば後述する図8のマネージャテーブルにドライバ（#1）に対応づけフックアドレスFSHook（）；を登録する。

【0022】S8は、S7に応答を受信したドライバ（#1）がS6の要求についてOKか判別する。YESの場合には、初期処理を完了したので終了する。NOの場合には、要求失敗と判断し、再試行あるいはエラーと

して中止などする。

【0023】以上によって、ドライバ4の初期処理が完了し、動作状態に設定されたこととなる。図3のフローチャートに示す順序に従い、図1の認証処理を詳細に説明する。

【0024】図3は、本発明のユーザ認証フローチャートを示す。図3において、S11は、アプリケーション1に対して、ユーザがユーザID、パスワードの入力を行う。

【0025】S12は、S11で入力されたユーザID、パスワードをもとにテーブルを参照してチェックを行い、OKか判別する。YESの場合には、S13に進む。NOの場合には、エラーとしてS11で再入力あるいはエラーとして中止する。

【0026】S13は、ユーザまたはアプリケーションからの暗号化／復号化フォルダ（ディレクトリ）の指定を行う。S14は、ドライバへの暗号化／復号化フォルダの指定を行う。

【0027】S15は、S14で通知を受けたドライバ（#1）が暗号化／復号化フォルダの設定・保存を行う。これは、例えば後述する図9に示すように、ユーザ毎に、フォルダ名に対応づけてリード処理／ライト処理毎に暗号化／復号化するものを登録する（○は暗号化／復号化する、×は暗号化／復号化しない）。

【0028】S16は、暗号化／復号化を開始指示する。S17は、ドライバ（#1）がS16の指示に対応して暗号化／復号化を開始する。

【0029】以上によって、認証されたユーザ毎にフォルダ毎に図9に示すようにリード処理およびライト処理について暗号化／復号化するか否かを登録し、当該登録に従い暗号化／復号化を開始する。

【0030】図4のフローチャートに示す順序に従い、図1のオープン処理を詳細に説明する。図4は、本発明のオープン処理フローチャートを示す。

【0031】図4において、S21は、アプリケーション1がオープン関数を発行する。S22は、ファイルシステム3がオープン処理を行う。S23は、ドライバ（#1）4が図示の下記のオープンイベントをOSより受信する（これは、ドライバの登録時に、OSに通知してもらうイベント、通知時にコールされる暗号化／復号化プログラムのアドレスを登録することで実現されている）。

【0032】・ファイル名（パス名を含む）：

- ・ファイル属性：
- ・ファイルハンドル：
- ・オープン関数のパラメタ：
- ・ファイルシステムへの復帰アドレス：
- ・その他

S24は、フォルダが暗号化／復号化対象かのチェックがOKか判別する。フォルダ名はパス名（ファイル名）

より抽出する。YESの場合（対象のファイルが存在するなどの場合）には、S27に進む。NOの場合には、暗号化／復号化対象外ファイルとしてS25で復帰アドレスへ復帰し、オープン処理に戻る。

【0033】S27は、ファイル属性から処理しないファイルを振り分ける。これは、図示の下記のファイル属性から処理する必要のないファイルを除外する。

- ・システムファイル：
- ・隠しファイル：
- ・ドライバファイル：
- ・その他

YESの場合には、暗号化／復号化の対象外のファイルと判明したので、S28で属性テーブルにファイルハンドルを保存し、S29に進む。一方、S27のNOの場合には、S29に進む。

【0034】S29は、復帰アドレスへ復帰する。S30は、オープン処理を完了する。S31は、オープン関数に復帰する。

【0035】以上の手順によって、アプリケーション1がオープン関数を発行し、ファイルのオープン処理を完了したこととなる。そして、後述する図5のリード処理あるいは図6のライト処理に進む。

【0036】図5のフローチャートに示す順序に従い、図1のリード処理を詳細に説明する。図5は、本発明のリード処理フローチャートを示す。

【0037】図5において、S41は、アプリケーション1がリード関数を発行する。S42は、ファイルシステム3がリード処理を行う。S43は、ドライバ（#1）4が図示の下記のリードイベントを受信する。

【0038】・ファイル名（パス名を含む）：

- ・ファイルハンドル：
- ・リードポイント：
- ・リードバッファ：
- ・リードサイズ：
- ・ファイルシステムへの復帰アドレス：
- ・その他

S44は、フォルダが暗号化／復号化対象かチェックがOKか判別する。YESの場合（暗号化／復号化対象の場合）には、S46に進む。NOの場合には、通常の処理としてS45で復帰アドレス（OS）へ復帰し、通常のリード処理を行う。

【0039】S46は、オープン処理時に更新した属性テーブルに当該要求のファイルハンドルがあるか否かをチェックする。これは、ファイルハンドルから暗号化／復号化の対象外のファイルか判別する。NOの場合には、S48に進む。YESの場合には、S47で復帰アドレスへ復帰し、通常のリード処理をする。

【0040】S48は、リード処理を呼び出す。これは、外部記憶装置10である例えばHDD（ハードディスク装置）のドライバを呼び出し、バウンダリ処理を行

ってデータをリードする。

【0041】S49は、S48で呼び出された例えばHDDドライバがHDDからバウンダリを考慮してデータを読み出す。S50は、S49で読み出したデータについて、同一サイズで復号化処理を行う。ここで、8バイト単位での暗号化処理、および端数処理を行う。これにより、8バイト単位に復号化処理を行い、読み出したデータの8バイトと、復号化した後のデータが8バイトとなるようにしており、復号化前と復号化後のデータが共に8バイトで同一とし、データの長さから両者が区別つかないようにし、セキュリティレベルを高めている。

【0042】S51は、リードサイズ分をリードバッファに設定する。S52は、復帰アドレスへ復帰する。S53は、リード処理を完了する。

【0043】S54は、リード関数に復帰する。以上によって、アプリケーション1がリード関数を発行すると、発行されたリード関数で指定されたデータがここでは、8バイト単位にファイルから読みだされて暗号化されているときは復号化し、暗号化されていないときは図示しないがそのままリードバッファに設定することをリード分のデータ分だけ繰り返し、アプリケーション1はリードバッファから読み出して一連のリード処理を行うことが可能となる。

【0044】図6のフローチャートに示す順序に従い、図1のライト処理を詳細に説明する。図6は、本発明のライト処理フローチャートを示す。

【0045】図6において、S61は、アプリケーション1がライト関数を発行する。S62は、ファイルシステム3がライト処理を行う。S63は、ドライバ（＃1）4が図示の下記のライトイベントを受信する。

【0046】・ファイル名（パス名を含む）：

・ファイルハンドル：

・ライトポイント：

・ライトバッファ：

・ライトサイズ：

・ファイルシステムへの復帰アドレス：

・その他

S64は、フォルダが暗号化／復号化対象かのチェックがOKか判別する。YESの場合（暗号化／復号化対象の場合）には、S66に進む。NOの場合には、S65で復帰アドレスへ復帰し、通常のライト処理をする。

【0047】S66は、オープン処理時に更新した属性テーブルに当該要求のファイルハンドルがあるか否かをチェックする。これは、ファイルハンドルから暗号化／復号化対象外のファイルか判別する。NOの場合には、S68に進む。YESの場合には、S67で復帰アドレスへ復帰し、通常のライト処理をする。

【0048】S68は、リード処理を呼び出す。これは、外部記憶装置10である例えばHDD（ハードディスク装置）のドライバを呼び出し、バウンダリ処理を行

ってデータをリードする。

【0049】S69は、S68で呼び出された例えばHDDドライバがHDDからバウンダリを考慮してデータを読み出す。S70は、S69で読み出したデータについて、同一サイズで復号化処理を行う。ここで、8バイト単位での復号化処理、および端数処理を行う。これにより、8バイト単位に復号化処理を行い、読み出したデータの8バイトと、復号化した後のデータが8バイトとなるようにしており、復号化前と復号化後のデータが共に8バイトで同一とし、データの長さから両者が区別つかないようにし、セキュリティレベルを高めている。

【0050】S71は、ライトデータを復号化データに上書きする。S72は、S71で上書きしたデータについて、同一サイズで暗号化処理を行う。ここで、8バイト単位での暗号化処理、および端数処理を行う。これにより、8バイト単位に暗号化処理を行い、上書きしたデータの8バイトと、暗号化した後のデータが8バイトとなるようにしており、暗号化前と暗号化後のデータが共に8バイトで同一とし、データの長さから両者が区別つかないようにし、セキュリティレベルを高めている。

【0051】S73は、ライト処理を呼び出す。これは、外部記憶装置10であるHDDのHDDドライバを呼び出し、バウンダリ処理を行ってデータをライトする。S74は、S73で呼び出されたHDDドライバがHDDからバウンダリを考慮してデータを書き戻す。

【0052】S75は、ライト処理を完了する。S76は、ドライバから復帰する。S77は、ライト関数に復帰する。

【0053】以上によって、アプリケーション1がライト関数を発行すると、発行されたライト関数で指定されたデータがここでは、8バイト単位にファイルから読みだされて暗号化されているときは復号化し、暗号化されていないときは図示しないがそのままとし、これらデータに対してライトデータを上書きした後、暗号化されていたときは暗号化したデータを、暗号化されていなかったときはそのままのデータを8バイト単位にファイルに書き戻すことを繰り返して一連のライト処理を行うことが可能となる。

【0054】図7のフローチャートに示す順序に従い、図1のクローズ処理を詳細に説明する。図7は、本発明のクローズ処理フローチャートを示す。

【0055】図7において、S81は、アプリケーション1がクローズ関数を発行する。S82は、ファイルシステム3がクローズ処理を行う。S83は、ドライバ（＃1）4が図示の下記のクローズイベントを受信する。

【0056】・ファイル名：

・ファイル属性：

・ファイルシステムへの復帰アドレス：

・その他

S84は、フォルダのチェックがOKか判別する。YESの場合(クローズ対象のファイルが存在するなどの場合)には、S86に進む。NOの場合には、S85で復帰アドレスへ復帰し、通常のクローズ処理をする。

【0057】S86は、オープン処理時に更新した属性テーブルに当該要求のファイルハンドルがあるか否かをチェックする。YESの場合には、S89に進む。NOの場合には、S87で復帰アドレスへ復帰し、通常のクローズ処理をする。

【0058】・システムファイル:

・隠しファイル:

・ドライバファイル:

・その他

S89は、属性テーブルに保存していたファイルハンドルを消去し、S90に進む。

【0059】S90は、復帰アドレスへ復帰する。S91は、クローズ処理を完了する。S92は、クローズ関数に復帰する。

【0060】以上の手順によって、アプリケーション1がクローズ関数を発行し、ファイルのクローズ処理を完了したこととなる。図8は、本発明のマネージャテーブル例を示す。これは、既述した図2の初期処理中のS7でOSに登録されたものであって、ここでは、図示の下記のようにドライバ毎にフックアドレスを登録し、データの授受を実現するためのものである。

【0061】ドライバ名      フックアドレス

ドライバ#1      フックアドレス    FSHook();  
図9は、本発明のユーザレベル管理テーブル例を示す。これは、ユーザ毎にフォルダ毎にリード処理およびライト処理の可否を登録するものであって、既述した図3のS15などで登録されるものである。このユーザレベル管理テーブルにより、ユーザ毎かつフォルダ毎に既述したリード処理およびライト処理の可否を参照し、既述した図5のリード処理、図6のライト処理を実行することにより、セキュリティを確保することが可能となる。

【0062】

【発明の効果】以上説明したように、本発明によれば、アクセス依頼に対応してリード手段6あるいはライト手段7が暗号化されているファイルからアクセス要求のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻す構成を採用しているため、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行うことができる。これらにより、

(1) 本願発明は、暗号化前と暗号化後、復号化前と

復号化後のデータの長さを同一(例えば8バイトで同一)としているため、ファイル内のデータが暗号化されているか否かをデータ長(ファイルサイズ)から判別できず、セキュリティレベルを高めることができる。

【0063】(2) 本願発明は、部分的(例えば8バイト単位)にファイルから暗号化データを読み出して復号化したり、部分的(例えば8バイト単位)にファイルから暗号化データを読み出して復号化しライトデータを上書きした後に暗号化してファイルに書き戻しているため、リード処理/ライト処理に必要な部分データのみをアクセスし、高速、迅速かつセキュリティレベルを高くできる。

【0064】(3) 従来のファイルから全データを作業ファイルに読み出して復号化したデータを、アプリケーションからアクセスするような場合には復号化したデータが作業ファイルに存在してデータの内容が判明してしまいセキュリティレベルが低い欠点があるが、本願発明では作業ファイルがなく、リード処理あるいはライト処理毎にその都度、ファイルからデータを部分的(例えば8バイト単位)に読み出して復号化したり、データを部分的(例えば8バイト単位)に読み出して復号化して上書きした後に暗号化してファイルに書き戻してしまい、セキュリティレベルが非常に高い。

【図面の簡単な説明】

【図1】本発明のシステムブロック図である。

【図2】本発明の初期処理フローチャートである。

【図3】本発明のユーザ認証フローチャートである。

【図4】本発明のオープン処理フローチャートである。

【図5】本発明のリード処理フローチャートである。

【図6】本発明のライト処理フローチャートである。

【図7】本発明のクローズ処理フローチャートである。

【図8】本発明のマネージャテーブル例である。

【図9】本発明のユーザレベル管理テーブル例である。

【符号の説明】

1: アプリケーション

2: ユーザ認証手段

3: ファイルシステム

4: ドライバ

5: オープン手段

6: リード手段

7: ライト手段

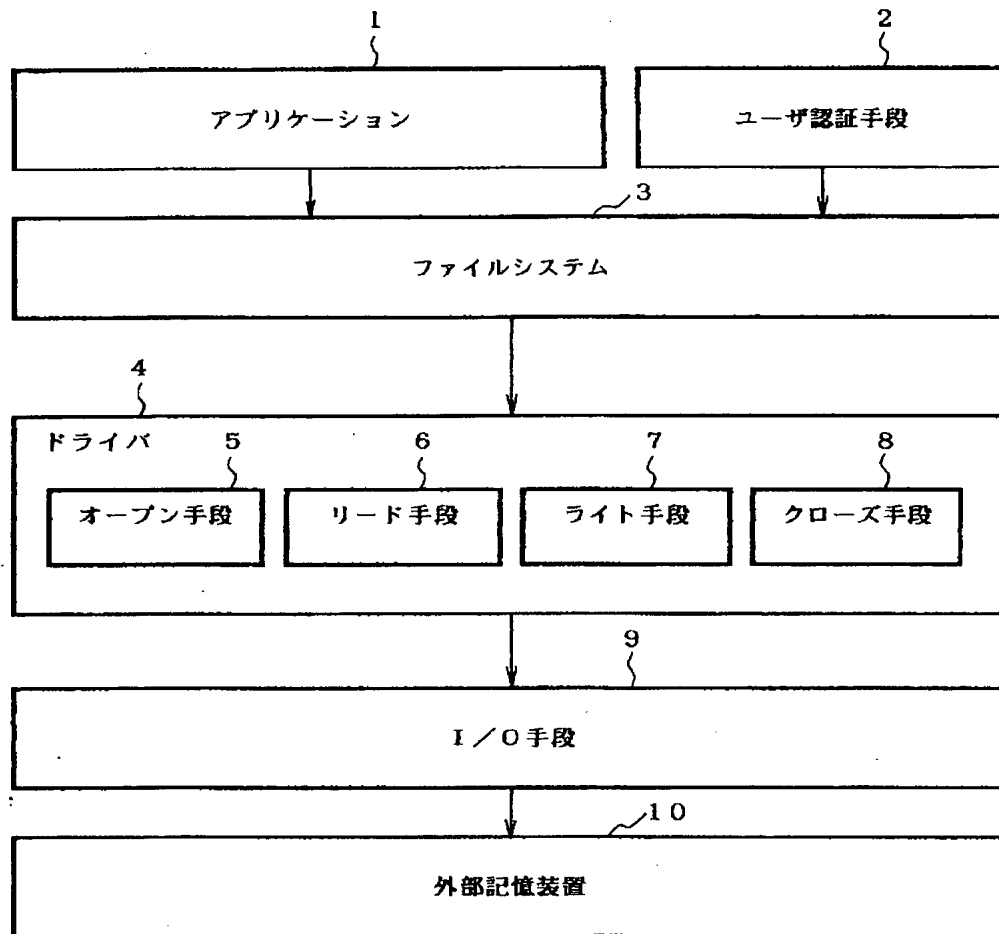
8: クローズ手段

9: I/O手段

10: 外部記憶装置

【図1】

本発明のシステムブロック図



【図8】

本発明のマネージャテーブル例

ドライバ#	フックアドレス	FSHook( ):
ドライバ#1		
ドライバ#	-	-
-	-	-

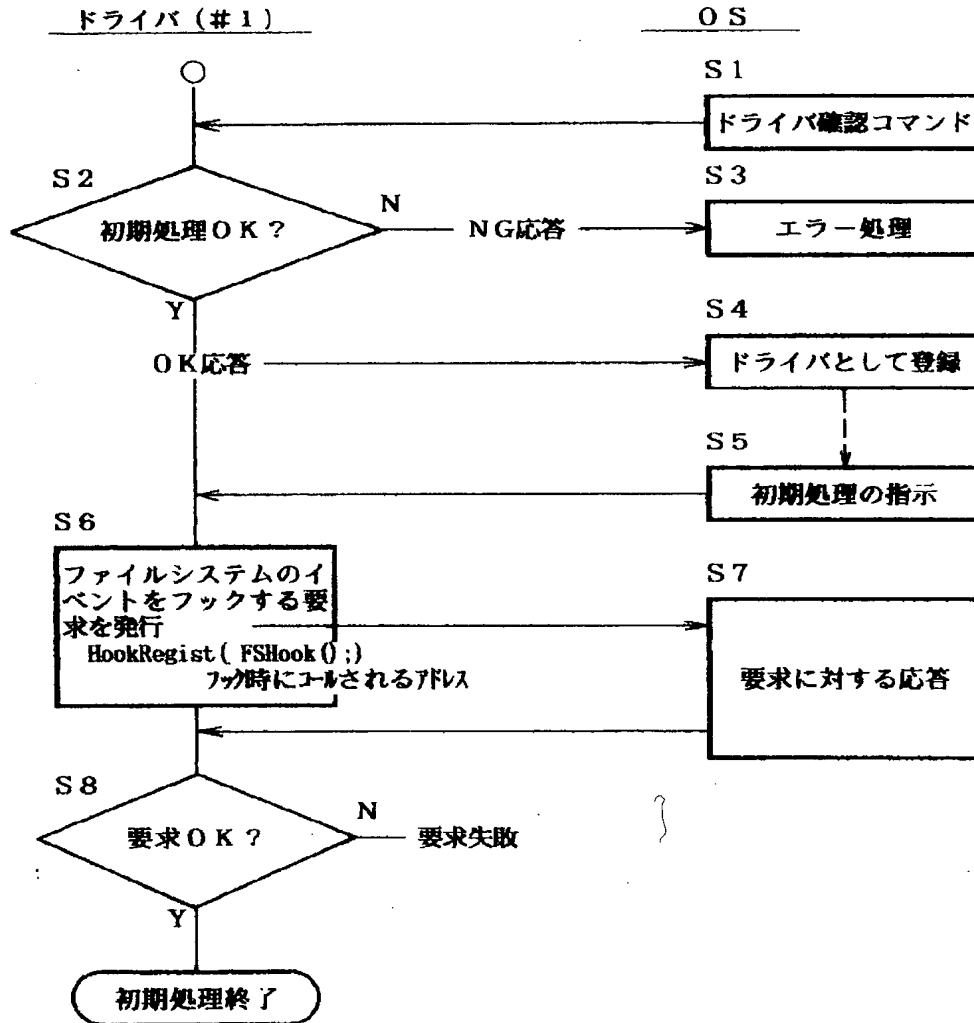
【図9】

本発明のユーザレバ管理テーブル例

フォルダ名	リード処理	ライト処理
フォルダA	○	○
フォルダB	○	×
フォルダC	×	○
フォルダD	×	×

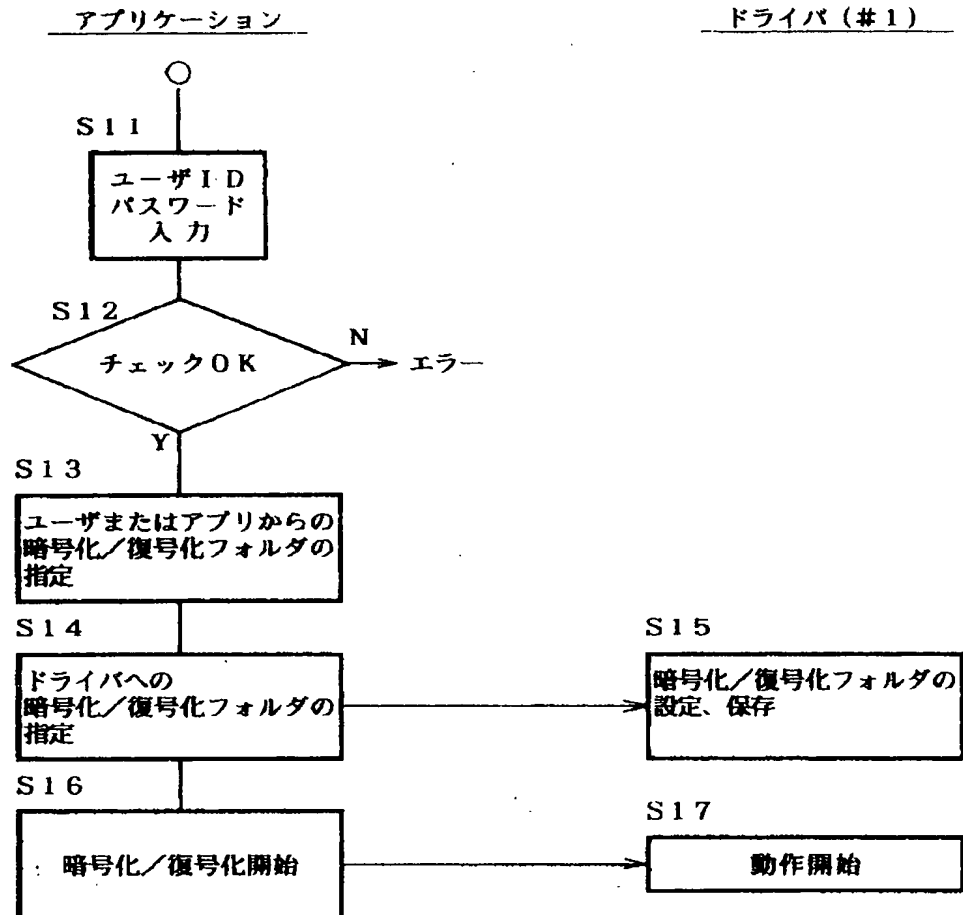


### 本発明の初期処理フローチャート



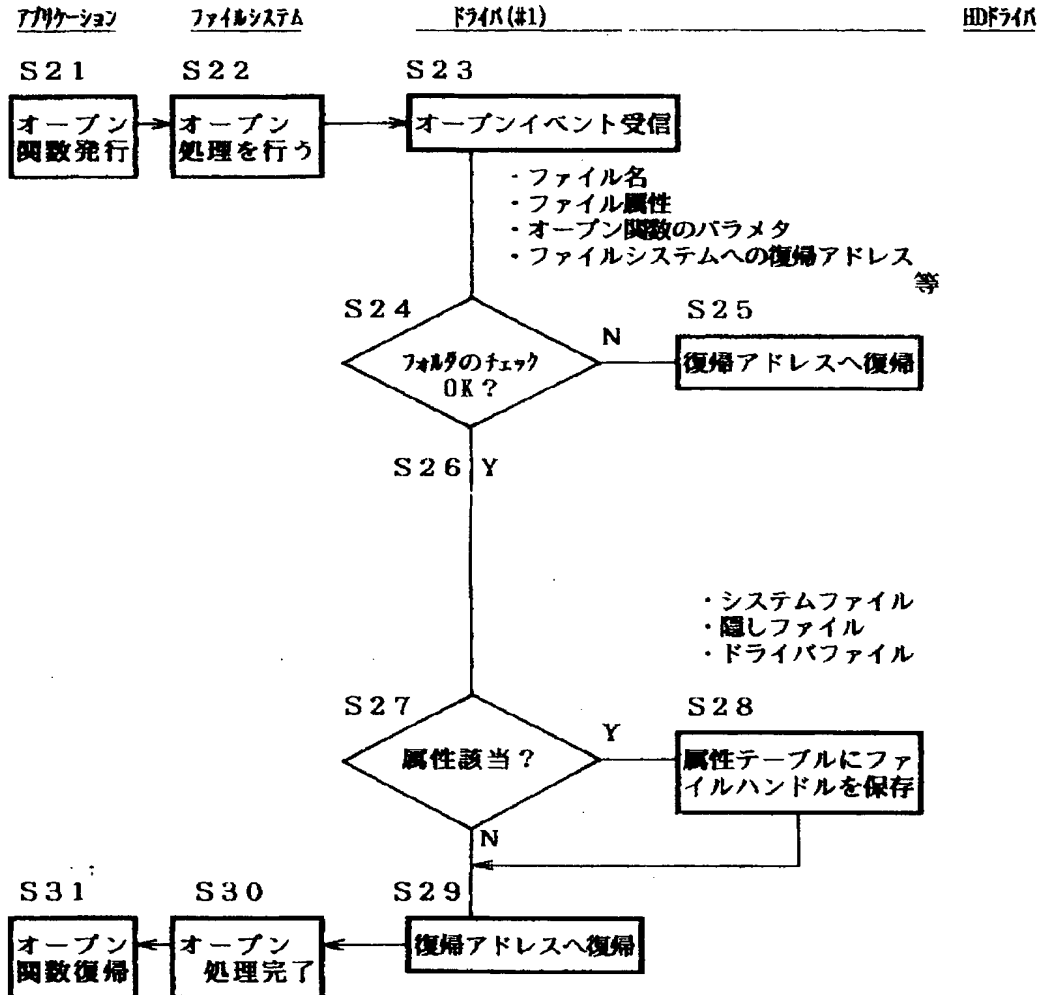
【図3】

## 本発明のユーザ認証フローチャート



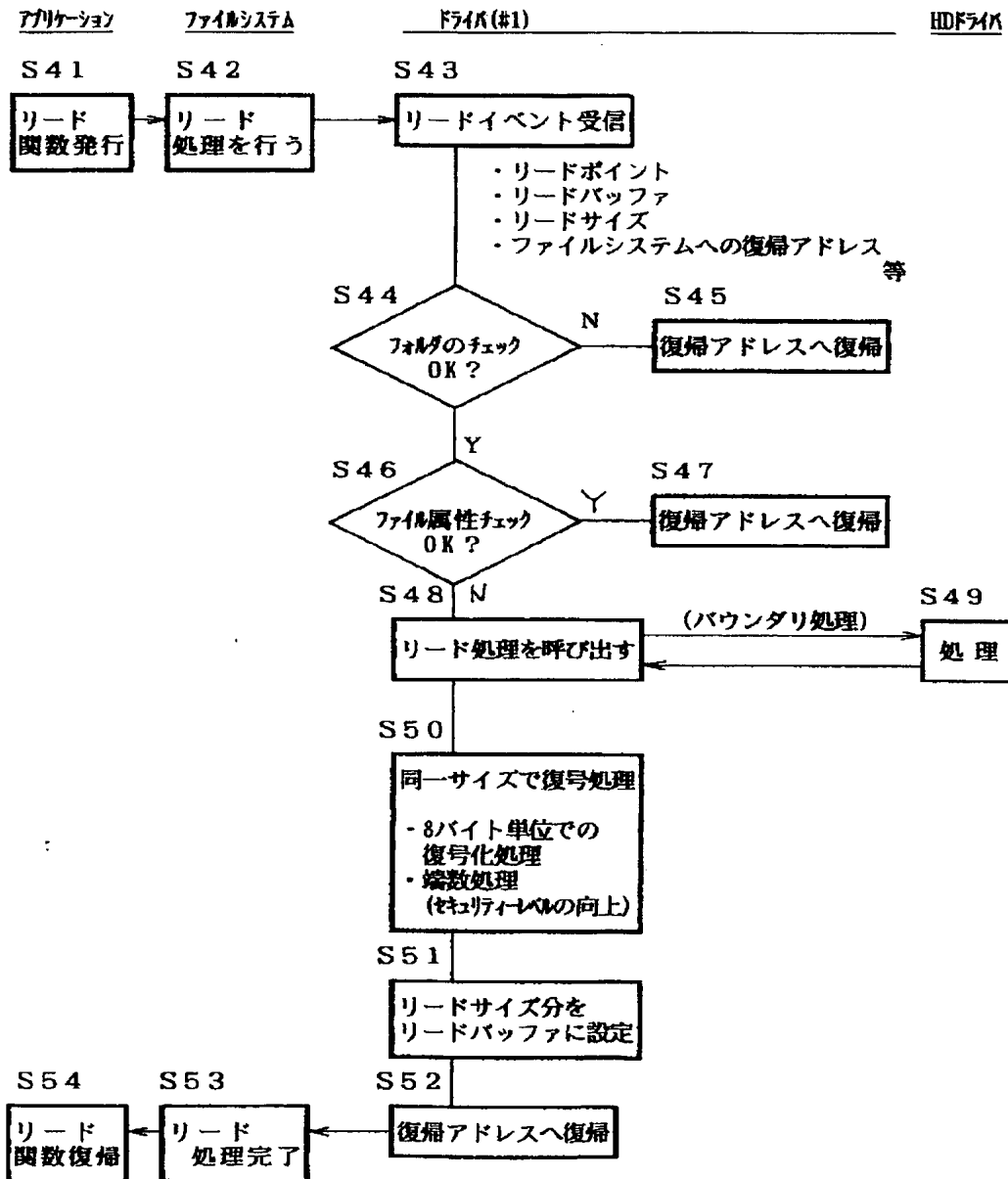
【図4】

## 本発明のオープン処理フローチャート



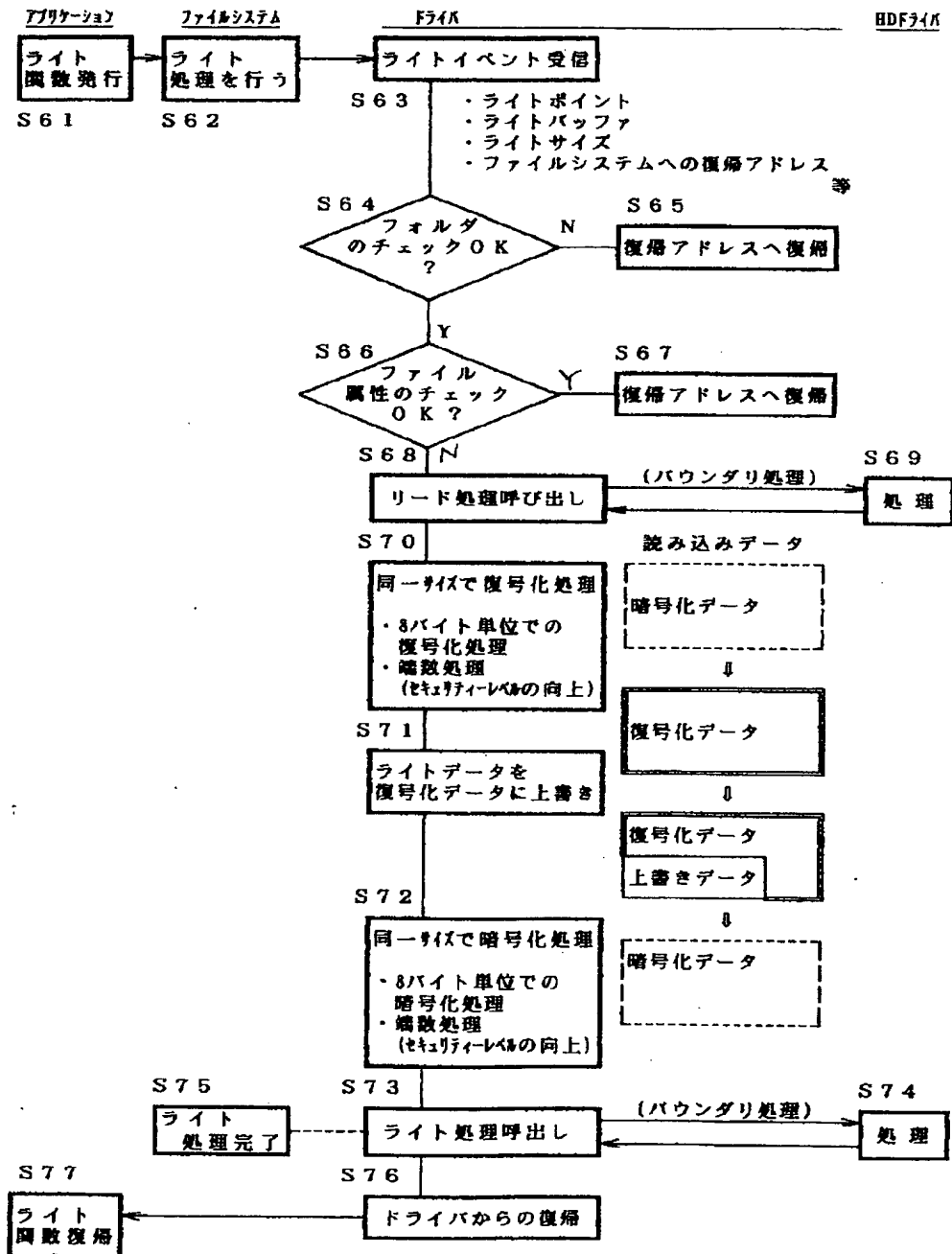
【図5】

## 本発明のリード処理フローチャート



【図6】

## 本発明のライト処理フローチャート



【図7】

## 本発明のクローズ処理フローチャート

